# 5、CC2530 外部中断控制 LED 开关

## 1．实验目的

1)、通过实验掌握 CC2530 芯片 GPIO 的配置方法，带你一步步走进嵌入式大门

2)、握 Led 驱动电路及开关 Led 的原理

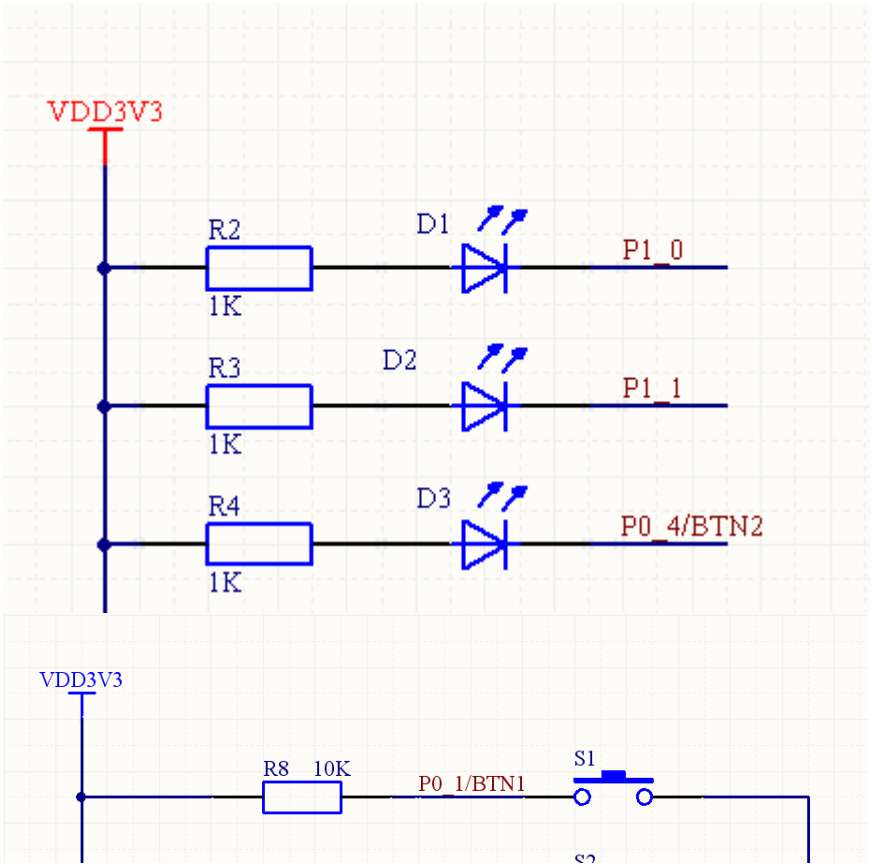
3)、过按键 S1 产生外部中断改变 LED1\LED2\LED3 状态

## 2．实验设备

硬件：PC 机一台 ZB 网关（底板、核心板、仿真器、USB 线） 一套

软件：2000/XP/win7 系统，IAR 8.10 集成开发环境

## 3.实验相关电路图




发光二极管是属于二极管的一种，具有二级管单向导电特性，即只有在正向电压（二极管的正极接正，负极接负）下才能导通发光。P1.0 引脚接发光二极管(D1)的负极,所以 P1.0 引脚输出低电平 D1 亮，P1.0 引脚输出高电平 D1 熄灭,D2,D3 同理。

按键 S1 接在 P0_1 上，当按键松开时，p0_1 通过电阻上拉为高电平，当按键 S1 按下时，p0_1 为低电平。

4．实验相关寄存器

操作 P1.0 我们需要掌握相关寄存器的作用和配置方法。如下表所示：

| 寄存器 | 作用 | 描述 |
|---|---|---|
| P1 (0x90) | 端口1 | 端口1。通用I / O端口。可以从SFR位寻址。 |
| P1SEL(0xF4) | 端口1 功能选择 | P1.7 到P0.0功能选择<br>0： 通用I / O<br>1： 外设功能 |
| P1DIR(0xFE) | 端口1 方向 | P1.7到P1.0的I/O方向<br>0： 输入<br>1： 输出 |
| P1INP(0xF6) | 端口1 输入模式 | P1.7到P1.2的I/O输入模式。由于P1.0 和P1.1 没有上拉/下拉功能，P1INP暂时不需要配置，了解一下为后面的实验打下基础<br>0： 上拉/下拉(见P2INP (0xF7) - 端口2输入模式)<br>1： 三态 |

CC2530 外部中断需要配置 P0IEN 、PICTL 、P0IFG、IEN1 寄存器。外部中断寄存器说明如下表所示：

| 寄存器 | 作用 | 描述 |
| --- | --- | --- |
| P0IEN(0xAB) | 端口0 中断屏蔽 | 端口P0.7到P0.0中断使能<br>0：中断禁用<br>1：中断使能。 |
| PICTL(0x8C) | 端口中断控制<br>P0ICON(Bit0) | 端口0，7到0输入模式下的中断配置。该位为所有端口0的输入选择中断请求条件。<br>0：输入的上升沿引起中断<br>1：输入的下降沿引起中断 |
| P0IFG(0x89) | 端口0 中断状态标志 | 端口0，位7到位0输入中断状态标志。当输入端口中断请求未决信号时，其相应的标志位将置1。 |
| IEN1 (0xB8) | 中断使能1<br>P0IE(Bit5) | 端口0中断使能<br>0：中断禁止<br>1：中断使能 |

按照表格寄存器的内容，对 P1.0 口进行配置，当 P1.0 输出低电平时 D1 被点亮，D2、D3 同理。S1 按下时 P0.1 产生外部中断从而控制 LED1/LED2/LED3 的亮灭所以配置如下：

P1SEL &=~0x01; //配置 P1.0 为通用 IO 口，默认为 0 的，可以不设
P1DIR |= 0x01; //P10 定义为输出

按键 S1 配置如下:

P0IEN |= 0x2; // P0.1 设置为中断方式 1：中断使能
PICTL |= 0x2; //下降沿触发
IEN1 |= 0x20; //允许 P0 口中断;
P0IFG = 0x00; //初始化中断标志位
EA = 1; //打开总中断

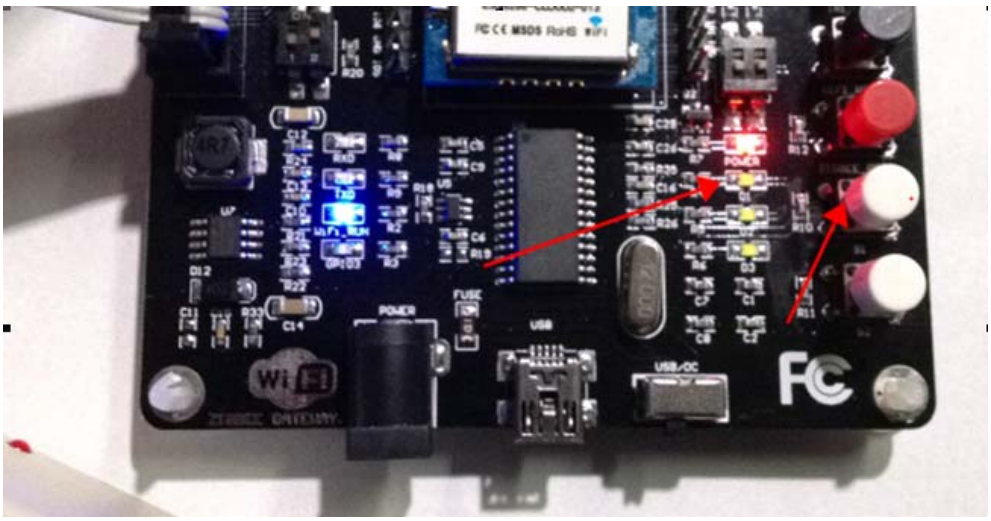
**5.源码分析**

```
/*************************************/
/*描述：按键 S1 外部中断方式改变 LED1 状态
*************************************/
#include <ioCC2530.h>

#define uint unsigned int
#define uchar unsigned char

//定义控制 LED 灯的端口
#define LED1 P1_0      //定义 LED1 为 P1.0 口控制
#define KEY1 P0_1      //中断口
//函数声明
void Delayms(uint);                //延时函数
void InitLed(void);          //初始化 P1 口
void KeyInit();              //按键初始化
uchar KeyValue=0;

/*****************************
//延时函数
*****************************/
void Delayms(uint xms)   //i=xms 即延时 i 毫秒
{
 uint i,j;
 for(i=xms;i>0;i--)
   for(j=587;j>0;j--);
}
/*****************************
LED 初始化程序
*****************************/
void InitLed(void)
```

```
{
  P1DIR |= 0x01; //P1_0 定义为输出
  LED1 = 1;      //LED1 灯熄灭
}

/*****************************
KEY 初始化程序--外部中断方式
*****************************/
void InitKey()
{
  P0IEN |= 0X2;  //P01 设置为中断方式
  PICTL |= 0X2; // 下降沿触发
  IEN1 |= 0X20;   // 允许 P0 口中断;
  P0IFG = 0x00;   // 初始化中断标志位
  EA = 1;
}

/*****************************
     中断处理函数
*****************************/
#pragma vector = P0INT_VECTOR    //格式：#pragma vector = 中断向量，紧接着是中断处理程序
  __interrupt void P0_ISR(void)
 {
  Delayms(10);          //去除抖动
  LED1=~LED1;           //改变 LED1 状态
  P0IFG = 0;           //清中断标志
  P0IF = 0;           //清中断标志
 }

/*****************************
```

```
        主函数
**************************/
void main(void)
{
        InitLed();              //调用初始化函数
        InitKey();
      while(1)
        {
      }
}
```

6.实验结果



按 S1  D1 亮 再次按 S1 D1 灭